

NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL

_S

Ps

NP

NP

\$G

\$O

NP

PA

_L

```

NN      NN  MM      MM      AAAAAA  FFFFFFFF  IIIIII  LL      EEEEEEEEE  SSSSSSSS
NN      NN  MM      MM      AAAAAA  FFFFFFFF  IIIIII  LL      EEEEEEEEE  SSSSSSSS
NN      NN  MMMM  MMMM  AA      AA  FF      LL      EE      SS
NN      NN  MMMM  MMMM  AA      AA  FF      LL      EE      SS
NNNN    NN  MM      MM  AA      AA  FF      LL      EE      SS
NNNN    NN  MM      MM  AA      AA  FF      LL      EE      SS
NN  NN  NN  MM      MM  AA      AA  FFFFFFFF  II      LL      EEEEEEE  SSSSSS
NN  NN  NN  MM      MM  AA      AA  FFFFFFFF  II      LL      EEEEEEE  SSSSSS
NN      NNNN  MM      MM  AAAAAAAAAA  FF      LL      EE      SS
NN      NNNN  MM      MM  AAAAAAAAAA  FF      LL      EE      SS
NN      NN  MM      MM  AA      AA  FF      LL      EE      SS
NN      NN  MM      MM  AA      AA  FF      LL      EE      SS
NN      NN  MM      MM  AA      AA  FF      IIIIII  LLLLLLLLL  EEEEEEEEE  SSSSSSSS
NN      NN  MM      MM  AA      AA  FF      IIIIII  LLLLLLLLL  EEEEEEEEE  SSSSSSSS
...

```

```
0001 0 %TITLE 'File Routines for Network Management'
0002 0 MODULE NMAFILES (
0003 0     LANGUAGE (BLISS32),
0004 0     ADDRESSING_MODE (NONEXTERNAL=GENERAL),
0005 0     ADDRESSING_MODE (EXTERNAL=GENERAL),
0006 0     IDENT = 'V04-000'
0007 0 ) =
0008 1 BEGIN
0009 1
0010 1
0011 1 *****
0012 1 *
0013 1 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0014 1 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0015 1 *   ALL RIGHTS RESERVED.
0016 1 *
0017 1 *   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0018 1 *   ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0019 1 *   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0020 1 *   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0021 1 *   OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0022 1 *   TRANSFERRED.
0023 1 *
0024 1 *   THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0025 1 *   AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0026 1 *   CORPORATION.
0027 1 *
0028 1 *   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0029 1 *   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0030 1 *
0031 1 *
0032 1 *****
0033 1
0034 1
0035 1 ++
0036 1 FACILITY:      DECnet Network Management Layer (NMA)
0037 1
0038 1 ABSTRACT:
0039 1
0040 1     This module contains routines which manage the files used by
0041 1     network management. These files contain permanent data about the
0042 1     configuration of the network.
0043 1
0044 1 ENVIRONMENT:  VAX/VMS Operating System
0045 1
0046 1 AUTHOR:      Darrell Duffy   , CREATION DATE: 18-December-1979
0047 1
0048 1 MODIFIED BY:
0049 1
0050 1     V03-007 MKP0007      Kathy Perko      2-April-1984
0051 1     If call is made to open a file and it is already open,
0052 1     do a $REWIND to get back to the beginning of the file.
0053 1
0054 1     V03-006 MKP0006      Kathy Perko      5-Feb-1984
0055 1     Fix NMA$READREC so that the correct key is returned to
0056 1     the caller.
0057 1
```


58	0058	1	V03-005	MKP0005	Kathy Perko	6-Aug-1983
59	0059	1		Enhance node permanent database to use multiple ISAM keys		
60	0060	1		so it's faster to access. When returning permanent database		
61	0061	1		records, don't include key in the data returned.		
62	0062	1				
63	0063	1	V03-004	MKP0004	Kathy Perko	25-April-1983
64	0064	1		Allow multiple NMLs to read and update the permanent database		
65	0065	1		files at once.		
66	0066	1				
67	0067	1	V03-004	MKP0004	Kathy Perko	25-April-1983
68	0068	1		Add NI configurator permanent database.		
69	0069	1				
70	0070	1	V03-003	MKP0003	Kathy Perko	12-Nov-1982
71	0071	1		Allow multiple NMLs to update the permanent database		
72	0072	1		files at once.		
73	0073	1				
74	0074	1	V03-002	MKP0002	Kathy Perko	18-Oct-1982
75	0075	1		Change the way NML opens and closes files so that it checks		
76	0076	1		to see if the operation has already been done. This will		
77	0077	1		improve the performance of operations which now open and close		
78	0078	1		various files more than once.		
79	0079	1				
80	0080	1	V03-001	MKP0001	Kathy Perko	3-Aug-1982
81	0081	1		Split module permanent data base into two: one for X25 and		
82	0082	1		one for X29.		
83	0083	1				
84	0084	1	V02-001	LMK0001	Len Kawell	27-Jul-1981
85	0085	1		Add CIRCUIT and MODULE files.		
86	0086	1				

```
88 0087 1 %SBTTL 'Definitions'
89 0088 1
90 0089 1
91 0090 1  TABLE OF CONTENTS:
92 0091 1
93 0092 1
94 0093 1  FORWARD ROUTINE
95 0094 1      NMA$OPENFILE,
96 0095 1      NMA$SELECTFILE,
97 0096 1      NMA$OPENFAB,
98 0097 1      NMA$CLOSEFILE,
99 0098 1      NMA$MATCHREC,
100 0099 1      NMA$READREC,
101 0100 1      NMA$WRITE REC,
102 0101 1      NMA$DELETEREC;
103 0102 1
104 0103 1
105 0104 1  INCLUDE FILES:
106 0105 1
107 0106 1
108 0107 1  LIBRARY 'LIB$:NMLLIB.L32';
109 0108 1  LIBRARY 'SHRLIB$:NMLIBRY.L32';
110 0109 1  LIBRARY 'SYS$LIBRARY:STARLET.L32';
111 0110 1
112 0111 1
113 0112 1  MACROS:
114 0113 1
115 0114 1
116 0115 1
117 0116 1  Define fields in a file descriptor.
118 0117 1
119 0118 1
120 0119 1  FIELD
121 0120 1      FDSCFLDS =
122 0121 1      SET
123 0122 1          FDSCFNS = [0, 0, 32, 0],
124 0123 1          FDSCFNA = [4, 0, 32, 0],
125 0124 1          FDSCFAB = [8, 0, 32, 0],
126 0125 1          FDSCRAB = [12, 0, 32, 0]
127 0126 1      TES;
128 0127 1
129 0128 1
130 0129 1  Macro to build file descriptors.
131 0130 1
132 0131 1      FILE      Designator of the file
133 0132 1      FILENAME  Filename string for file
134 0133 1
135 0134 1
136 0135 1  MACRO
137 M 0136 1      $NMA_BLDFILEDSC [FILE, FILENAME] = ! Build as many as you like
138 M 0137 1
139 M 0138 1      OWN      ! Declare the fab and rab
140 M 0139 1          $NAME ('NMA$_', FILE, '_FAB') : $FAB_DECL,
141 M 0140 1          $NAME ('NMA$_', FILE, '_RAB') : $RAB_DECL;
142 M 0141 1
143 M 0142 1      BIND      ! The descriptor
144 M 0143 1          $NAME ('NMA$_', FILE, '_DSC') =
```

```

145 M 0144 1      UPLIT
146 M 0145 1      (
147 M 0146 1      %CHARCOUNT (FILENAME),      ! Descriptor of filename str
148 M 0147 1      UPLIT BYTE (FILENAME),      ! Addr
149 M 0148 1      %NAME ('NMA$A_', FILE, '_FAB'), ! Fab address
150 M 0149 1      %NAME ('NMA$A_', FILE, '_RAB'), ! Rab address
151 M 0150 1      );
152 M 0151 1      %:
153 M 0152 1      !
154 M 0153 1      !
155 M 0154 1      ! EQUATED SYMBOLS:
156 M 0155 1      !
157 M 0156 1      !
158 M 0157 1      !
159 M 0158 1      ! OWN STORAGE:
160 M 0159 1      !
161 M 0160 1      !
162 M 0161 1      OWN
163 M 0162 1      NMA$W_KEYBUF : WORD;      ! Key buffer
164 M 0163 1      !
165 P 0164 1      $NMA_BLD$FILEDSC
166 P 0165 1      ?
167 P 0166 1      NODE,      'NETNODE',      ! Remote node database
168 P 0167 1      LINE,      'NETLINE',      ! Line database
169 P 0168 1      LOG,      'NETLOGING',      ! Logging database
170 P 0169 1      OBJ,      'NETOBJECT',      ! Object database
171 P 0170 1      CIR,      'NETCIRC',      ! Circuit database
172 P 0171 1      X25,      'NETX25',      ! X25 Module database
173 P 0172 1      X29,      'NETX29',      ! X29 Module database
174 P 0173 1      CNF,      'NETCONF',      ! Ni Configurator Module database
175 M 0174 1      );
176 M 0175 1      !
177 M 0176 1      !
178 M 0177 1      ! EXTERNAL REFERENCES:
179 M 0178 1      !
180 M 0179 1      !
181 M 0180 1      EXTERNAL ROUTINE
182 M 0181 1      NML$DEBUG_MSG,
183 M 0182 1      NML$DEBUG_TXT,
184 M 0183 1      NML$LOGFICEOP,
185 M 0184 1      NML$LOGRECORDOP;
186 M 0185 1
```



```
188 0186 1 %SBTTL 'NMA$OPENFILE Open a specified file'
189 0187 1 GLOBAL ROUTINE NMA$OPENFILE (FILEID, ACCESS) =
190 0188 1
191 0189 1 ++
192 0190 1 FUNCTIONAL DESCRIPTION:
193 0191 1
194 0192 1 This routine opens a specified file for specified access.
195 0193 1 The fileid specifies the file, or all files and the access
196 0194 1 specifies read only or read write.
197 0195 1
198 0196 1 FORMAL PARAMETERS:
199 0197 1
200 0198 1 FILEID Value of the fileid parameter (NMA$C_OPN_XXXXX)
201 0199 1 ACCESS Value of the access parameter (NMA$C_OPN_AC_Rx)
202 0200 1
203 0201 1 ROUTINE VALUE:
204 0202 1 COMPLETION CODES:
205 0203 1
206 0204 1 Failure or RMS error
207 0205 1
208 0206 1 --
209 0207 1
210 0208 2 BEGIN
211 0209 2
212 0210 2 LOCAL
213 0211 2 FAB : REF BLOCK [1, BYTE], ! The fab for the file
214 0212 2 FILEDSC : REF BLOCK [1, BYTE] ! File descriptor
215 0213 2 FIELD (FDSCFLDS),
216 0214 2 RAB, ! The rab for the file
217 0215 2 STATUS; ! Status return
218 0216 2
219 0217 2 IF .FILEID EQL NMA$C_OPN_ALL THEN ! If ALL
220 0218 2 BEGIN
221 0219 3
222 0220 3 INCRU IDX FROM NMA$C_OPN_MIN ! Open all the files by
223 0221 3 TO NMA$C_OPN_MAX DO ! Calling ourselves
224 0222 4 BEGIN
225 0223 4 STATUS = NMA$OPENFILE (.IDX, .ACCESS); ! Call ourself to open it
226 0224 4 IF NOT .STATUS THEN
227 0225 4 EXITLOOP;
228 0226 4 END
229 0227 3 END
230 0228 2 ELSE
231 0229 3 BEGIN
232 0230 3 STATUS = NMA$ SUCCESS;
233 0231 3 IF NMA$SELECTFILE (.FILEID, FILEDSC) THEN ! Obtain descriptor address
234 0232 4 BEGIN
235 0233 4 FAB = .FILEDSC [FDSCFAB]; ! Get address of FAB
236 0234 4 IF .FAB [FAB$W_IF1] EQL 0 THEN ! If file isn't open, do it.
237 0235 5 BEGIN
238 0236 5 STATUS = NMA$OPENFAB (.FILEDSC, .ACCESS); ! Open file by descriptor
239 0237 5 IF .STATUS THEN
240 0238 5 NML$LOGFILEOP (DBG$C_FILEIO,
241 0239 5 .FILEID,
242 0240 5 $ASCID ('file opened.'));
243 0241 5 END
244 0242 4 ELSE
```

```

245 0243 4
246 0244 4
247 0245 4
248 0246 4
249 0247 5
250 0248 5
251 0249 5
252 0250 4
253 0251 4
254 0252 3
255 0253 3
256 0254 2
257 0255 2
258 0256 2
259 0257 1

```

The file is already open, so don't reopen it. However,
 set RMS's "next record" back to the beginning of the file.

```

BEGIN
RAB = .FILEDSC [FDSCRAB];
$REWIND (RAB = .RAB);
END;
! Point to the rab

ELSE
RETURN NMA$_BADFID;
END;
! If not all, return failure

RETURN .STATUS
END;

```

```

.TITLE NMAFILES File Routines for Network Management
.IDENT \V04-000\

.PSECT $PLITS$,NOWRT,NOEXE,2

45 44 4F 4E 54 45 4E 00000 P.AAB: .ASCII \NETNODE\
00007 .BLKB 1
00008 P.AAA: .LONG 7
0000C .ADDRESS P.AAB, NMA$_NODE_FAB, NMA$_NODE_RAB
45 4E 49 4C 54 45 4E 00018 P.AAD: .ASCII \NETLINE\
0001F .BLKB 1
00020 P.AAC: .LONG 7
00024 .ADDRESS P.AAD, NMA$_LINE_FAB, NMA$_LINE_RAB
47 4E 49 47 4F 4C 54 45 4E 00030 P.AAF: .ASCII \NETLOGING\
00039 .BLKB 3
0003C P.AAE: .LONG 9
00040 .ADDRESS P.AAF, NMA$_LOG_FAB, NMA$_LOG_RAB
54 43 45 4A 42 4F 54 45 4E 0004C P.AAH: .ASCII \NETOBJECT\
00055 .BLKB 3
00058 P.AAG: .LONG 9
0005C .ADDRESS P.AAH, NMA$_OBJ_FAB, NMA$_OBJ_RAB
43 52 49 43 54 45 4E 00068 P.AAJ: .ASCII \NETCIRC\
0006F .BLKB 1
00070 P.AAI: .LONG 7
00074 .ADDRESS P.AAJ, NMA$_CIR_FAB, NMA$_CIR_RAB
35 32 58 54 45 4E 00080 P.AAL: .ASCII \NETX25\
00086 .BLKB 2
00088 P.AAK: .LONG 3
0008C .ADDRESS P.AAL, NMA$_X25_FAB, NMA$_X25_RAB
39 32 58 54 45 4E 00098 P.AAN: .ASCII \NETX29\
0009E .BLKB 2
000A0 P.AAM: .LONG 6
000A4 .ADDRESS P.AAN, NMA$_X29_FAB, NMA$_X29_RAB
46 4E 4F 43 54 45 4E 000B0 P.AAP: .ASCII \NETCONF\
000B7 .BLKB 1
000B8 P.AAO: .LONG 7
000BC .ADDRESS P.AAP, NMA$_CNF_FAB, NMA$_CNF_RAB
2E 64 65 6E 65 70 6F 20 65 6C 69 66 000C8 P.AAR: .ASCII \file opened.\
000D4 P.AAQ: .LONG 12
000D8 .ADDRESS P.AAR

```


.PSECT \$OWNS,NOEXE,2

00000 NMA\$W_KEYBUF:
 .BKLB 2
00002 .BKLB 2
00004 NMA\$A_NODE_FAB:
 .BKLB 80
00054 NMA\$A_NODE_RAB:
 .BKLB 68
00098 NMA\$A_LINE_FAB:
 .BKLB 80
000E8 NMA\$A_LINE_RAB:
 .BKLB 68
0012C NMA\$A_LOG_FAB:
 .BKLB 80
0017C NMA\$A_LOG_RAB:
 .BKLB 68
001C0 NMA\$A_OBJ_FAB:
 .BKLB 80
00210 NMA\$A_OBJ_RAB:
 .BKLB 68
00254 NMA\$A_CIR_FAB:
 .BKLB 80
002A4 NMA\$A_CIR_RAB:
 .BKLB 68
002E8 NMA\$A_X25_FAB:
 .BKLB 80
00338 NMA\$A_X25_RAB:
 .BKLB 68
0037C NMA\$A_X29_FAB:
 .BKLB 80
003CC NMA\$A_X29_RAB:
 .BKLB 68
00410 NMA\$A_CNF_FAB:
 .BKLB 80
00460 NMA\$A_CNF_RAB:
 .BKLB 68

NMA\$A_NODE_DSC= P.AAA
NMA\$A_LINE_DSC= P.AAC
NMA\$A_LOG_DSC= P.AAE
NMA\$A_OBJ_DSC= P.AAG
NMA\$A_CIR_DSC= P.AAI
NMA\$A_X25_DSC= P.AAK
NMA\$A_X29_DSC= P.AAM
NMA\$A_CNF_DSC= P.AAO
.EXTRN NML\$DEBUG MSG, NML\$DEBUG TXT
.EXTRN NML\$LOGFICEOP, NML\$LOGRECORDOP
.EXTRN SYSSREWIND

.PSECT \$CODE\$,NOWRT,2

.ENTRY NMA\$OPENFILE, Save R2,R3
SUBL2 #4, SP
CMPL FILEID, #127
BNEQ 2\$

0000007F SE
8F

04

000C 00000
04 C2 00002
AC D1 00005
1A 12 0000D

: 0187
:
:
: 0217
:

			52	D4	0000F	CLRL	IDX		0223	
		08	AC	DD	00011	1\$:	PUSHL	ACCESS		
			52	DD	00014		PUSHL	IDX		
E6	AF		02	FB	00016		CALLS	#2, NMA\$OPENFILE		
	53		50	DO	0001A		MOVL	R0, STATUS		
	5A		53	E9	0001D		BLBC	STATUS, 4\$	0224	
			52	D6	00020		INCL	IDX	0220	
	07		52	D1	00022		CMPL	IDX, #7		
			EA	1B	00025		BLEQU	1\$		
			51	11	00027		BRB	4\$	0218	
	53		01	DO	00029	2\$:	MOVL	#1, STATUS	0230	
			5E	DD	0002C		PUSHL	SP	0231	
		04	AC	DD	0002E		PUSHL	FILEID		
00000000V	00		02	FB	00031		CALLS	#2, NMA\$SELECTFILE		
	43		50	E9	00038		BLBC	R0, 5\$		
	50		6E	DO	0003B		MOVL	FILEDSC, R0	0233	
	51		08	A0	DO	0003E	MOVL	8(R0), FAB		
			02	A1	B5	00042	TSTW	2(FAB)	0234	
			26	12	00045		BNEQ	3\$		
		08	AC	DD	00047		PUSHL	ACCESS	0236	
			50	DD	0004A		PUSHL	R0		
00000000V	00		02	FB	0004C		CALLS	#2, NMA\$OPENFAB		
	53		50	DO	00053		MOVL	R0, STATUS		
	21		53	E9	00056		BLBC	STATUS, 4\$	0237	
		00000000'	00	9F	00059		PUSHAB	P.AAQ	0240	
		04	AC	DD	0005F		PUSHL	FILEID	0239	
			01	DD	00062		PUSHL	#1	0238	
00000000G	00		03	FB	00064		CALLS	#3, NML\$LOGFILEOP		
			0D	11	0006B		BRB	4\$	0234	
	50		0C	A0	DO	0006D	3\$:	MOVL	12(R0), RAB	0248
			50	DD	00071		PUSHL	RAB	0249	
00000000G	00		01	FB	00073		CALLS	#1, SYSS\$REWIND		
	50		53	DO	0007A	4\$:	MOVL	STATUS, R0	0256	
				04	0007D		RET			
			50	D4	0007E	5\$:	CLRL	R0	0257	
			04	00080			RET			

; Routine Size: 129 bytes, Routine Base: \$CODE\$ + 0000

```
261 0258 1 %SBTTL 'NMASELECTFILE Return a file descriptor'
262 0259 1 GLOBAL ROUTINE NMASELECTFILE (FILEID, FILEDSC) =
263 0260 1
264 0261 1 ++
265 0262 1 FUNCTIONAL DESCRIPTION:
266 0263 1
267 0264 1 This routine returns the address of the file descriptor for a
268 0265 1 specified file. Failure is returned if the fileid is not
269 0266 1 valid.
270 0267 1
271 0268 1 FORMAL PARAMETERS:
272 0269 1
273 0270 1 FILEID Value of the fileid (NMASC_OPN_xxxxx)
274 0271 1 FILEDSC Address to return address of file descriptor
275 0272 1
276 0273 1 IMPLICIT INPUTS:
277 0274 1
278 0275 1 NONE
279 0276 1
280 0277 1 IMPLICIT OUTPUTS:
281 0278 1
282 0279 1 NONE
283 0280 1
284 0281 1 ROUTINE VALUE:
285 0282 1 COMPLETION CODES:
286 0283 1
287 0284 1 Success or failure
288 0285 1
289 0286 1 SIDE EFFECTS:
290 0287 1
291 0288 1 NONE
292 0289 1
293 0290 1 --
294 0291 1
295 0292 2 BEGIN
296 0293 2
297 0294 2 LOCAL
298 0295 2 STATUS;
299 0296 2
300 0297 2 STATUS = NMAS_SUCCESS;
301 0298 2
302 0299 2 .FILEDSC = ! Obtain the file descriptor
303 0300 2 BEGIN ! Address
304 0301 2
305 0302 2 CASE .FILEID FROM NMASC_OPN_MIN TO NMASC_OPN_MAX OF
306 0303 2 SET
307 0304 2
308 0305 2 [NMASC_OPN_NODE]: NMASA_NODE_DSC;
309 0306 2 [NMASC_OPN_LINE]: NMASA_LINE_DSC;
310 0307 2 [NMASC_OPN_LOG]: NMASA_LOG_DSC;
311 0308 2 [NMASC_OPN_OBJ]: NMASA_OBJ_DSC;
312 0309 2 [NMASC_OPN_CIR]: NMASA_CIR_DSC;
313 0310 2 [NMASC_OPN_X25]: NMASA_X25_DSC;
314 0311 2 [NMASC_OPN_X29]: NMASA_X29_DSC;
315 0312 2 [NMASC_OPN_CNF]: NMASA_CNF_DSC;
316 0313 2 [INRANGE
317 0314 2 OUTRANGE]: ! Code not known, fail
```


NMAFILES
V04-000

File Routines for Network Management
NMA\$SELECTFILE Return a file descriptor

H 16
16-Sep-1984 00:42:37
14-Sep-1984 12:50:02

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NML.SRC]NMAFILES.B32;1
Page 10
(4)

```

0315      BEGIN
0316
0317      STATUS = NMA$_BADFID;
0318      0                                ! Return invalid descriptor
0319
0320      END;
0321
0322      TES
0323      END;
0324      RETURN .STATUS
0325
0326      END;
0327

```

0025	001F	0019	0014	0004	00000	.ENTRY	NMA\$SELECTFILE, Save R2	0259
003F	0038	0031	002B	00019	00002	MOVAB	NMA\$A_NODE_DSC, R2	0297
					00009	MOVL	#1, STATUS	0302
					0000C	CASEL	FILEID, #0, #7	
					00011	.WORD	2\$-1\$,-	
					00019		3\$-1\$,-	
							4\$-1\$,-	
							5\$-1\$,-	
							6\$-1\$,-	
							7\$-1\$,-	
							8\$-1\$,-	
							9\$-1\$,-	
							NO	0315
						CLRD	10\$	0302
						BRB	NMA\$A_NODE_DSC, R0	
						BRB	10\$	
						MOVAB	NMA\$A_LINE_DSC, R0	
						BRB	10\$	
						MOVAB	NMA\$A_LOG_DSC, R0	
						BRB	10\$	
						MOVAB	NMA\$A_OBJ_DSC, R0	
						BRB	10\$	
						MOVAB	NMA\$A_CIR_DSC, R0	
						BRB	10\$	
						MOVAB	NMA\$A_X25_DSC, R0	
						BRB	10\$	
						MOVAB	NMA\$A_X29_DSC, R0	
						BRB	10\$	
						MOVAB	NMA\$A_CNF_DSC, R0	
						MOVL	R0, @FILEDSC	0300
						MOVL	STATUS, R0	0325
						RET		0327

; Routine Size: 93 bytes, Routine Base: \$CODE\$ + 0081

```
332 0328 1 XSBTTL 'NMA$OPENFAB Open or Create a File'
333 0329 1 ROUTINE NMA$OPENFAB (FILEDSC, ACCESS) =
334 0330 1
335 0331 1 ++
336 0332 1 FUNCTIONAL DESCRIPTION:
337 0333 1
338 0334 1 This routine does the actual open or create of a file.
339 0335 1 First the fab is loaded with the correct attributes and then
340 0336 1 a create or open service is done. Create is used if the file
341 0337 1 is to be opened with read-write access and the FOP CIF bit is
342 0338 1 specified so that the file is created if it does not exist.
343 0339 1 The created file will be indexed with a two byte binary key.
344 0340 1 A rather large bucket size is used to allow for long records.
345 0341 1 The protection is set to be read for world and group and the
346 0342 1 UIC is set to the system.
347 0343 1
348 0344 1 FORMAL PARAMETERS:
349 0345 1
350 0346 1 FILEDSC Address of the filedescrptor for the file
351 0347 1 ACCESS Value of the access parameter
352 0348 1
353 0349 1 IMPLICIT INPUTS:
354 0350 1
355 0351 1 NONE
356 0352 1
357 0353 1 IMPLICIT OUTPUTS:
358 0354 1
359 0355 1 NONE
360 0356 1
361 0357 1 ROUTINE VALUE:
362 0358 1 COMPLETION CODES:
363 0359 1
364 0360 1 Success or an RMS error
365 0361 1
366 0362 1 SIDE EFFECTS:
367 0363 1
368 0364 1 NONE
369 0365 1
370 0366 1 --
371 0367 1
372 0368 2 BEGIN
373 0369 2
374 0370 2 MAP ! File descriptor format
375 0371 2 FILEDSC : REF BLOCK [1, BYTE] FIELD (FDSCFLDS);
376 0372 2
377 0373 2 LOCAL
378 0374 2 STATUS, ! Return status
379 0375 2 FAB, ! Fab address
380 0376 2 RAB, ! Rab address
381 0377 2 FNS, ! Filename size
382 0378 2 FNA; ! Filename address
383 0379 2
384 0380 2 OWN
385 0381 2 KEYXAB : $XABKEY_DECL, ! Key xab for create
386 0382 2 PROXAB : $XABPRO_DECL; ! Protection xab for create
387 0383 2
388 0384 2 FNA = .FILEDSC [FDSCFNA]; ! Obtain descriptor fields
```

```
389 0385 2 FNS = .FILEDSC [FDSCFNS];
390 0386 FAB = .FILEDSC [FDSCFAB];
391 0387 RAB = .FILEDSC [FDSCRAB];
392 0388
393 0389 IF .ACCESS EQL NMA$C_OPN_AC_RW ! Check access for read write
394 0390 THEN
395 0391 BEGIN
396 0392 $FAB_INIT ! Initialize fab for create
397 0393 (
398 0394 FAB = .FAB, ! Fab address
399 0395 BKS = 9, ! Bucket size
400 0396 DNM = 'SYS$SYSTEM:.DAT', ! Default filename string
401 0397 FAC = (UPD, PUT, GET, DEL), ! File access
402 0398 FNA = .FNA, ! Filename string address
403 0399 FNS = .FNS, ! Filename string size
404 0400 FOP = (CIF, MXV), ! File open codes (create if, max ver)
405 0401 ORG = IDX, ! Organization
406 0402 RFM = VAR, ! Record format
407 0403 SHR = (UPD, PUT, GET, DEL), ! Share
408 0404 XAB = PROXAB ! Xab chain
409 0405 );
410 0406
411 0407 $XABKEY_INIT ! Initialize key xab
412 0408 (
413 0409 XAB = KEYXAB, ! Xab address
414 0410 DTP = BN2, ! 2 byte binary
415 0411 POSO = 0, ! Position
416 0412 SIZO = 2, ! Size
417 0413 KREF = 0 ! Key reference (primary)
418 0414 );
419 0415
420 0416 $XABPRO_INIT ! Initialize protection xab
421 0417 (
422 0418 XAB = PROXAB, ! Xab address
423 0419 UIC = (1, 4), ! Uic of owner (system)
424 0420 PRO = (RWED, RWED, .), ! Protection (group and world no access)
425 0421 NXT = KEYXAB ! Chain
426 0422 );
427 0423
428 0424 STATUS = $CREATE (FAB = .FAB); ! Create the file if not found
429 0425
430 0426 END
431 0427
432 0428 ELSE
433 0429 BEGIN
434 0430 $FAB_INIT ! Initialize the fab
435 0431 (
436 0432 FAB = .FAB, ! Fab address
437 0433 FAC = (GET), ! File access
438 0434 FNA = .FNA, ! Filename string address
439 0435 FNS = .FNS, ! Filename string size
440 0436 DNM = 'SYS$SYSTEM:.DAT', ! Default filename string
441 0437 SHR = (UPD, PUT, GET, DEL) ! Share
442 0438 );
443 0439
444 0440
445 0441
```


004C	8F	00	1D A6	20 90 0003C	MOVB	#32, 29(FAB)	
			1F A6	02 90 0004J	MOVB	#2, 31(FAB)	
			24 A6	6A 9E 00044	MOVAB	PROXAB, 36(FAB)	
			2C A6	58 D0 00048	MOVL	FNA, 44(FAB)	
			30 A6 00000000'	00 9E 0004C	MOVAB	P.AAS, 48(FAB)	
			34 A6	59 90 00054	MOVB	FNS, 52(FAB)	
			35 A6	0F 90 00058	MOVB	#15, 53(FAB)	
			3E A6	09 90 0005C	MOVB	#9, 62(FAB)	
			6E	00 2C 00060	MOVCS	#0, (SP), #0, #76, \$RMS_PTR	0415
			B4 AA	AA 00067			
			4C15	8F B0 00069	MOVW	#19477, \$RMS_PTR	
			B4 AA	02 90 0006F	MOVB	#2, \$RMS_PTR+19	
0058	8F	00	C7 AA	02 90 00073	MOVB	#2, \$RMS_PTR+46	
			E2 A6	00 2C 00077	MOVCS	#0, (SP), #0, #88, \$RMS_PTR	0423
			6E	6A 0007E			
			6A 5813	8F B0 0007F	MOVW	#22547, \$RMS_PTR	
			04 AA	AA 9E 00084	MOVAB	KEYXAB, \$RMS_PTR+4	
			08 AA	8F B0 00089	MOVW	#-256, \$RMS_PTR+8	
			0C AA 00010004	8F D0 0008F	MOVL	#65540, \$RMS_PTR+12	
			00000000G	56 DD 00097	PUSHL	FAB	0425
			00	01 FB 00099	CALLS	#1, SYS\$CREATE	
0050	8F	00	6E	34 11 000A0	BRB	3\$	0389
			66 5003	00 2C 000A2	MOVCS	#0, (SP), #0, #80, (FAB)	0440
			0F02	66 000A9			
			16 A6	8F B0 000AA	MOVW	#20483, (FAB)	
			1F A6	8F B0 000AF	MOVW	#3842, 22(FAB)	
			2C A6	02 90 000B5	MOVB	#2, 31(FAB)	
			30 A6 00000000'	58 D0 000B9	MOVL	FNA, 44(FAB)	
			34 A6	00 9E 000BD	MOVAB	P.AAT, 48(FAB)	
			35 A6	59 90 000C5	MOVB	FNS, 52(FAB)	
			00000000G	0F 90 000C9	MOVB	#15, 53(FAB)	
			00	56 DD 000CD	PUSHL	FAB	0442
			30	01 FB 000CF	CALLS	#1, SYS\$OPEN	
0044	8F	00	6E	50 E9 000D6	BLBC	STATUS, 4\$	0446
			67 4401	00 2C 000D9	MOVCS	#0, (SP), #0, #68, (RAB)	0459
			04 A7 00200010	67 000E0			
			1E A7	8F B0 000E1	MOVW	#17409, (RAB)	
			30 A7	8F D0 000E6	MOVL	#2097168, 4(RAB)	
			34 A7	01 90 000EE	MOVB	#1, 30(RAB)	
			3C A7	CA 9E 000F2	MOVAB	NMA\$W KEYBUF, 48(RAB)	
			00000000G	02 90 000F8	MOVB	#2, 52(RAB)	
			00	56 D0 000FC	MOVL	FAB, 60(RAB)	
				57 DD 00100	PUSHL	RAB	0461
				01 FB 00102	CALLS	#1, SYS\$CONNECT	
				04 00109	RET		0463

; Routine Size: 266 bytes, Routine Base: \$CODE\$ + 00DE

```
469 0464 1 %SBTTL 'NMA$CLOSEFILE Close a specified file'
470 0465 1 GLOBAL ROUTINE NMA$CLOSEFILE (FILEID) =
471 0466 1
472 0467 1 !++
473 0468 1 FUNCTIONAL DESCRIPTION:
474 0469 1
475 0470 1 This routine closes a specified file or all the files.
476 0471 1
477 0472 1 FORMAL PARAMETERS:
478 0473 1
479 0474 1 FILEID Value of the fileid parameter (NMA$OPN_xxxxx)
480 0475 1
481 0476 1 ROUTINE VALUE:
482 0477 1 COMPLETION CODES:
483 0478 1
484 0479 1 Status of last close operation.
485 0480 1
486 0481 1 !--
487 0482 1
488 0483 2 BEGIN
489 0484 2
490 0485 2 LOCAL
491 0486 2 FAB : REF BLOCK [1, BYTE], ! The fab for the file
492 0487 2 FILEDSC : REF BLOCK [1, BYTE] ! File descriptor
493 0488 2 FIELD (FDSCFLDS),
494 0489 2 STATUS; ! Status return
495 0490 2
496 0491 2 STATUS = NMA$ SUCCESS;
497 0492 2 IF NMA$SELECTFILE (.FILEID, FILEDSC) THEN ! Obtain descriptor address
498 0493 3 BEGIN
499 0494 3 FAB = .FILEDSC [FDSCFAB]; ! Get address of FAB
500 0495 3 IF .FAB [FAB$W_IF1] NEQ 0 THEN ! If file isn't closed, do it.
501 0496 4 BEGIN
502 0497 4 STATUS =
503 0498 4 $CLOSE (FAB = .FILEDSC [FDSCFAB]); ! Call RMS to close the file
504 0499 4 IF .STATUS THEN
505 0500 4 NML$LOGFILEOP (DBG$C FILEID,
506 0501 4 FILEID,
507 0502 4 $ASCID ('file closed.'));
508 0503 3 END;
509 0504 3 END
510 0505 2 ELSE
511 0506 2 STATUS = NMA$_BADFID;
512 0507 2 RETURN .STATUS
513 0508 2
514 0509 1 END;
```

```
2E 64 65 73 6F 6C 63 20 65 6C 69 66 000FA P.AAV: .PSECT $SPLITS, NOWRT, NOEXE, 2
00106 .ASCII \file closed.\
0000000C 00108 P.AAU: .BLKB 2
00000000 0010C .LONG 12
ADDRESS P.AAV
.EXTRN SYS$CLOSE
```


			0004 000C0	.PSECT	\$CODE\$,NOWRT,2	
	SE		04 C2 00002	.ENTRY	NMA\$CLOSEFILE, Save R2	: 0465
	52		01 D0 00005	SUBL2	#4, SP	: 0491
			5E DD 00008	MOVL	#1, STATUS	: 0492
		04	AC DD 0000A	PUSHL	SP	
FE87	CF		02 FB 0000D	PUSHL	FILEID	
	30		50 E9 00012	CALLS	#2, NMA\$SELECTFILE	
	50		6E D0 00015	BLBC	R0, 1\$: 0494
	51	08	A0 D0 00018	MOVL	FILEDSC, R0	
		02	A1 B5 0001C	MOVL	8(R0), FAB	: 0495
			26 13 0001F	TSTW	2(FAB)	
		08	A0 DD 00021	BEQL	2\$: 0498
00000000G	00		01 FB 00024	PUSHL	8(R0)	
	52		50 D0 00028	CALLS	#1, SYS\$CLOSE	
	16		52 E9 0002E	MOVL	R0, STATUS	: 0499
		00000000'	00 9F 00031	BLBC	STATUS, 2\$: 0502
		04	AC DD 00037	PUSHAB	P.AAU	: 0501
			01 DD 0003A	PUSHL	FILEID	: 0500
00000000G	00		03 FB 0003C	PUSHL	#1	
			02 11 00043	CALLS	#3, NML\$LOGFILEOP	: 0492
			52 D4 00045	BRB	2\$: 0506
	50		52 D0 00047	CLRL	STATUS	: 0507
			04 0004A	MOVL	STATUS, R0	: 0509
				RET		

; Routine Size: 75 bytes, Routine Base: \$CODE\$ + 01E8

```
516 0510 1 $SBTTL 'NMA$MATCHREC Find a Record in a File'
517 0511 1 GLOBAL ROUTINE NMA$MATCHREC (FILEID, BUFDSC, KEYADR, FIELD CODE,
518 0512 1 FIELD SIZE, FIELDADR, RTNDSC) =
519 0513 1
520 0514 1 **
521 0515 1 FUNCTIONAL DESCRIPTION:
522 0516 1
523 0517 1 This routine searches a database for a record containing a given
524 0518 1 field containing given data. Degenerate cases are provided for
525 0519 1 returning all records, or all records containing a specific field.
526 0520 1
527 0521 1 FORMAL PARAMETERS:
528 0522 1
529 0523 1 FILEID Value of the fileid code (NMA$C.OPN_xxxxx)
530 0524 1 BUFDSC Address of a descriptor of a buffer to use
531 0525 1 KEYADR Address of a word containing the key to start reading
532 0526 1 Key value is returned in this word.
533 0527 1 FIELD CODE Value of the field code (zero for wildcard)*****
534 0528 1 FIELD SIZE Value of the field size (zero for wildcard)
535 0529 1 FIELDADR Address of the field data
536 0530 1 RTNDSC Address of a descriptor to return descriptor of data
537 0531 1
538 0532 1 IMPLICIT INPUTS:
539 0533 1
540 0534 1 NONE
541 0535 1
542 0536 1 IMPLICIT OUTPUTS:
543 0537 1
544 0538 1 NONE
545 0539 1
546 0540 1 ROUTINE VALUE:
547 0541 1 COMPLETION CODES:
548 0542 1
549 0543 1 NMA or RMS error status
550 0544 1
551 0545 1 SIDE EFFECTS:
552 0546 1
553 0547 1 NONE
554 0548 1
555 0549 1 --
556 0550 1
557 0551 1 BEGIN
558 0552 1
559 0553 1 MAP
560 0554 1 BUFDSC : REF VECTOR, ! Buffer to use for record
561 0555 1 RTNDSC : REF VECTOR; ! Return data descriptor
562 0556 1
563 0557 1 LOCAL
564 0558 1 FILEDSC : REF BLOCK [1, BYTE] ! File descriptor
565 0559 1 FIELD (FDSC$FLDS)
566 0560 1 RAB : REF BLOCK [1, BYTE], ! The rab for the file
567 0561 1 LCLDSC : VECTOR [2], ! A local data descriptor
568 0562 1 FAB : REF BLOCK [, BYTE], ! The fab for the file
569 0563 1 FLDADR, ! Field address
570 0564 1 FLDSIZ, ! Field size
571 0565 1 STATUS; ! Status return
572 0566 1
```

```
573 0567 2 EXTERNAL ROUTINE
574 0568 NMASSEARCHFLD; ! Search for a field value
575 0569
576 0570 STATUS = NMASSELECTFILE (.FILEID,
577 0571 FILEDSC); ! Obtain the file descriptor
578 0572
579 0573 IF NOT .STATUS
580 0574 THEN
581 0575 RETURN .STATUS; ! Bogus fileid
582 0576
583 0577 RAB = .FILEDSC [FDSCRAB]; ! Point to the rab
584 0578 FAB = .FILEDSC [FDSCFAB]; ! Get address of FAB
585 0579
586 0580 IF .FAB [FABSW_IFI] EQL 0 ! If file not open,
587 0581 THEN
588 0582 RETURN .FAB [FABSL_STS]; ! return open failure status
589 0583
590 0584 RAB [RABSW_USZ] = .BUFDSC [0]; ! Set the buffer to use
591 0585 RAB [RABSL_UBF] = .BUFDSC [1];
592 0586
593 0587 NMASW_KEYBUF = ..KEYADR; ! And the key value to use
594 0588
595 0589 WHILE 1 ! Try this forever
596 0590 DO
597 0591 BEGIN
598 0592
599 0593 STATUS = $GET (RAB = .RAB); ! Read a record
600 0594
601 0595 LCLDSC [0] = .RAB [RABSW_RSZ]; ! Pickup the real record descriptor
602 0596 LCLDSC [1] = .RAB [RABSL_RBF];
603 0597 RTNDSC [0] = .RAB [RABSW_RSZ] - NML$K_PERM_KEYS_LEN;
604 0598 RTNDSC [1] = .RAB [RABSL_RBF] + NML$K_PERM_KEYS_LEN;
605 0599
606 0600 IF NOT .STATUS ! If no good, return
607 0601 THEN
608 0602 RETURN .STATUS;
609 0603
610 0604 NMASW_KEYBUF = ! Set the keyvalue returned
611 0605 .(.LCLDSC [1]) <0, 16, 0>;
612 0606
613 0607 (.KEYADR) <0, 16, 0> = .NMASW_KEYBUF; ! Return for user to remember
614 0608
615 0609 FLDADR = 0; ! Start search from beginning
616 0610 IF NMASSEARCHFLD ! Look for the field
617 0611 (
618 0612 .RTNDSC, ! Here is the data
619 0613 .FIELD$CODE, ! Value of the code to look for
620 0614 .FLDSIZ, ! Return the size here
621 0615 .FLDADR ! Return the address here
622 0616 )
623 0617 THEN
624 0618 BEGIN
625 0619
626 0620 IF .FIELD$SIZE EQL 0 ! Wildcard
627 0621 THEN
628 0622 BEGIN
629 0623
```



```

630      0624      STATUS = NMAS_SUCCESS; ! It always succeeds
631      0625      EXITLOOP;
632      0626
633      0627      END;
634      0628
635      0629      IF CH$EQL                      ! Look at the data
636      0630      (
637      0631      .FLDSIZ,                      ! Data in record
638      0632      .FLDADR,
639      0633      .FIELD$IZE,                  ! User data
640      0634      .FIELDADR,
641      0635      0
642      0636      )
643      0637      THEN
644      0638      BEGIN
645      0639
646      0640      STATUS = NMAS_SUCCESS; ! We found such a record
647      0641      EXITLOOP;
648      0642
649      0643      END;
650      0644      END;
651      0645
652      0646      NMASW KEYBUF = .NMASW KEYBUF + 1; ! Increment key ****
653      0647      (.KEYADR) < 0, 16, 0> = .NMASW_KEYBUF; ! Return for user to remember
654      0648
655      0649      END;
656      0650
657      0651      IF .STATUS
658      0652      THEN
659      0653      NML$LOGRECORDOP (DBG$C FILEIO,
660      0654      .FILEID,
661      0655      $ASCID ('record matched'),
662      0656      LC$LDC);
663      0657
664      0658      RETURN .STATUS
665      0659
666      0660      END;

```

.PSECT SPLITS,NOWRT,NOEXE,2

```

64 65 68 63 74 61 60 20 64 72 6F 63 65 72 00110 P.AAX:
                                0011E
                                0000000E 00120 P.AAW:
                                00000000 00124

```

```
.ASCII \record matched\  
.BLKB 2  
.LONG 14  
.ADDRESS P.AAX
```

```
.EXTRN NMASSEARCHFLD, SYSSGET
```

.PSECT \$CODES,NGWRT,2

			00FC	00000
57	000000000'	00	9E	00002
5E		14	C2	00009
		5E	DD	0000C
	04	AC	DD	0000E
FE38	CF	02	FB	00011

```

.ENTRY  NMMATCHREC, Save R2,R3,R4,R5,R6,R7
MOVAB  NMA$W KEYBUF, R7
SUBL2  #20, SP
PUSHL  SP
PUSHL  FILEID
CALLS  #2, NMA$SELECTFILE

```

0511

0570

		56		50	D0	00016	MOVL	R0, STATUS		
		4F		56	E9	00019	BLBC	STATUS, 3\$	0573	
		50		6E	D0	0001C	MOVL	FILEDSC, R0	0577	
		54	0C	A0	D0	0001F	MOVL	12(R0), RAB		
		50	08	A0	D0	00023	MOVL	8(R0), FAB	0578	
			02	A0	B5	00027	TSTW	2(FAB)	0580	
				05	12	0002A	BNEQ	1\$		
		50	08	A0	D0	0002C	MOVL	8(FAB), R0	0582	
				04	00	00030	RET			
		50	08	AC	D0	00031	1\$: MOVL	BUFDSC, R0	0584	
20		A4		60	B0	00035	MOVW	(R0), 32(RAB)		
24		A4	04	A0	D0	00039	MOVL	4(R0), 36(RAB)	0585	
		67	0C	BC	B0	0003E	MOVW	@KEYADR, NMASW_KEYBUF	0587	
		55	1C	AC	D0	00042	MOVL	RTNDSC, R5	0598	
				54	DD	00046	2\$: PUSHL	RAB	0593	
	00000000G	00		01	FB	00048	CALLS	#1, SYSSGET		
		56		50	D0	0004F	MOVL	R0, STATUS		
	0C	AE	22	A4	3C	00052	MOVZWL	34(RAB), LCLDSC	0595	
	10	AE	28	A4	D0	00057	MOVL	40(RAB), LCLDSC+4	0596	
	1C	BC	22	A4	3C	0005C	MOVZWL	34(RAB), @RTNDSC	0597	
	1C	BC		02	C2	00061	SUBL2	#2, @RTNDSC		
04	AS	28		02	C1	00065	ADDL3	#2, 40(RAB), 4(R5)	0598	
		57		56	E9	0006B	3\$: BLBC	STATUS, 7\$	0600	
		67	10	BE	B0	0006E	MOVW	@LCLDSC+4, NMASW_KEYBUF	0605	
	0C	BC		67	B0	00072	MOVW	NMASW_KEYBUF, @KEYADR	0607	
			04	AE	D4	00076	CLRL	FLDADR	0609	
			04	AE	9F	00079	PUSHAB	FLDADR	0611	
			0C	AE	9F	0007C	PUSHAB	FLDSIZ		
			10	AC	DD	0007F	PUSHL	FIELD CODE	0613	
			1C	AC	DD	00082	PUSHL	RTNDSC	0612	
	00000000G	00		04	FB	00085	CALLS	#4, NMASSEARCHFLD		
		16		50	E9	0008C	BLBC	R0, 5\$		
			14	AC	D5	0008F	TSTL	FIELD SIZE	0620	
				0C	13	00092	BEQL	4\$		
14	AC		00	04	BE	08	CMPC5	FLDSIZ, @FLDADR, #0, FIELD SIZE, @FIELDADR	0630	
			18	BC		0009C				
				05	12	0009E	BNEQ	5\$		
		56		01	D0	000A0	4\$: MOVL	#1, STATUS	0640	
				08	11	000A3	BRB	6\$	0638	
				67	B6	000A5	5\$: INCW	NMASW_KEYBUF	0646	
	0C	BC		67	B0	000A7	MOVW	NMASW_KEYBUF, @KEYADR	0647	
				99	11	000AB	BRB	2\$	0589	
		15		56	E9	000AD	6\$: BLBC	STATUS, 7\$	0651	
			0C	AE	9F	000B0	PUSHAB	LCLDSC	0653	
			00000000	00	9F	000B3	PUSHAB	P.AAW	0655	
			04	AC	DD	000B9	PUSHL	FILEID	0654	
				01	DD	000BC	PUSHL	#1	0653	
	00000000G	00		04	FB	000BE	CALLS	#4, NML\$LOGRECORDOP		
		50		56	DD	000C5	7\$: MOVL	STATUS, R0	0658	
				04	00	000C8	RET		0660	

; Routine Size: 201 bytes, Routine Base: \$CODE\$ + 0233

```
668 0661 1 ZSBTTL 'NMA$READREC Get a record from a File'
669 0662 1 GLOBAL ROUTINE NMA$READREC (FILEID, KEYADR, BUFDSC, RTNDSC) =
670 0663 1
671 0664 1 !++
672 0665 1 FUNCTIONAL DESCRIPTION:
673 0666 1
674 0667 1 This routine reads the next database record starting at the specified
675 0668 1 key.
676 0669 1
677 0670 1 FORMAL PARAMETERS:
678 0671 1
679 0672 1 FILEID Value of the fileid code (NMASC_OPN_xxxxx)
680 0673 1 KEYADR Address of a word containing the key to start reading
681 0674 1 Key value is returned in this word.
682 0675 1 BUFDSC Address of a descriptor of a buffer to use
683 0676 1 RTNDSC Address of a descriptor to return descriptor of data
684 0677 1
685 0678 1 IMPLICIT INPUTS:
686 0679 1
687 0680 1 NONE
688 0681 1
689 0682 1 IMPLICIT OUTPUTS:
690 0683 1
691 0684 1 NONE
692 0685 1
693 0686 1 ROUTINE VALUE:
694 0687 1 COMPLETION CODES:
695 0688 1
696 0689 1 NMA or RMS error status
697 0690 1
698 0691 1 SIDE EFFECTS:
699 0692 1
700 0693 1 NONE
701 0694 1
702 0695 1 --
703 0696 1
704 0697 1 BEGIN
705 0698 1
706 0699 1 MAP
707 0700 1 BUFDSC : REF VECTOR, ! Buffer to use for record
708 0701 1 RTNDSC : REF VECTOR; ! Return data descriptor
709 0702 1
710 0703 1 LOCAL
711 0704 1 FILEDSC : REF BLOCK [1, BYTE] ! File descriptor
712 0705 1 FIELD (FDSCFLDS),
713 0706 1 FAB : REF BLOCK [1, BYTE], ! The fab for the file
714 0707 1 RAB : REF BLOCK [1, BYTE], ! The rab for the file
715 0708 1 LCLDSC : VECTOR [2],
716 0709 1 STATUS; ! Status return
717 0710 1
718 0711 1 STATUS = NMA$SELECTFILE (.FILEID,
719 0712 1 FILEDSC); ! Obtain the file descriptor
720 0713 1
721 0714 1 IF NOT .STATUS
722 0715 1 THEN
723 0716 1 RETURN .STATUS; ! Bogus fileid
724 0717 1
```



```
725 0718 2
726 0719 2 RAB = .FILEDSC [FDSCRAB]; ! Point to the rab
727 0720 2 FAB = .FILEDSC [FDSCFAB]; ! Get address of FAB
728 0721 2
729 0722 2 IF .FAB [FAB$J_IF1] EQL 0 ! If file not open,
730 0723 2 THEN
731 0724 2 RETURN .FAB [FAB$L_STS]; ! Return open failure status
732 0725 2
733 0726 2 RAB [RAB$W_USZ] = .BUFDSC [0]; ! Set the buffer to use
734 0727 2 RAB [RAB$L_UBF] = .BUFDSC [1];
735 0728 2
736 0729 2 NMA$W_KEYBUF = ..KEYADR; ! And the key value to use
737 0730 2
738 0731 2 STATUS = $GET (RAB = .RAB); ! Read a record
739 0732 2
740 0733 2 RTNDSC [0] = .RAB [RAB$W_RSZ] - NML$K_PERM_KEYS_LEN;
741 0734 2 RTNDSC [1] = .RAB [RAB$L_RBF] + NML$K_PERM_KEYS_LEN;
742 0735 2
743 0736 2 IF NOT .STATUS ! If no good, return
744 0737 2 THEN
745 0738 2 RETURN .STATUS;
746 0739 2
747 0740 2 LCLDSC [0] = .RAB [RAB$W_RSZ];
748 0741 2 LCLDSC [1] = .RAB [RAB$L_RBF];
749 0742 2
750 0743 2 (.KEYADR)<0,16,0> = (.LCLDSC [1])<0,16>; ! Return for user to remember
751 0744 2
752 0745 2 NML$LOGRECORDOP (DBG$C_FILEIO,
753 0746 2 .FILEID,
754 0747 2 $ASCII ('record read'),
755 0748 2 LCLDSC);
756 0749 2
757 0750 2 RETURN NMA$_SUCCESS
758 0751 2
759 0752 2 END;
```

```
64 61 65 72 20 64 72 6F 63 65 72 00128 P.AAZ: .ASCII \record read\
00133 .BLKB 1
0000000B 00134 P.AAY: .LONG 11
00000000 00138 .ADDRESS P.AAZ
```

```
.PSECT $PLITS$,NOWRT,NOEXE,2
```

```
0004 00000
5E 0C C2 00002
04 5E DD 00005
FD76 CF AC DD 00007
6A 02 FB 0000A
51 50 E9 0000F
51 6E D0 00012
08 A1 7D 00015
```

```
.PSECT $CODE$,NOWRT,2
```

```
.ENTRY NMA$READREC, Save R2
SUBL2 #12, SP
PUSHL SP
PUSHL FILEID
CALLS #2, NMA$SELECTFILE
BLBC STATUS, 2$
MOVL FILEDSC, R1
MOVQ 8(R1), FAB
```

```
: 0662
: 0711
: 0714
: 0719
: 0720
```

		02	A1	B5	00019	TSTW	2(FAB)	..	0722	
		05	12	0001C		BNEQ	1\$			
	50	08	A1	D0	0001E	MOVL	8(FAB), R0	..	0724	
				04	00022	RET				
	51	0C	AC	D0	00023	1\$: MOVL	BUFDSC, R1	..	0726	
20	A2		61	B0	00027	MOVW	(R1), 32(RAB)			
24	A2	04	A1	D0	0002B	MOVL	4(R1), 36(RAB)	..	0727	
00000000	00	08	BC	B0	00030	MOVW	@KEYADR, NMA\$W_KEYBUF	..	0729	
			52	DD	00038	PUSHL	RAB	..	0731	
00000000G	00		01	FB	0003A	CALLS	#1, SYS\$GET			
	51	10	AC	D0	00041	MOVL	RTNDSC, R1	..	0733	
	61	22	A2	3C	00045	MOVZWL	34(RAB), (R1)			
	61		02	C2	00049	SUBL2	#2, (R1)			
04	A1	28	A2	02	C1	0004C	ADDL3	#2, 40(RAB), 4(R1)	..	0734
	27		50	E9	00052	BLBC	STATUS, 2\$..	0736	
04	AE	22	A2	3C	00055	MOVZWL	34(RAB), LCLDSC	..	0740	
08	AE	28	A2	D0	0005A	MOVL	40(RAB), LCLDSC+4	..	0741	
08	BC	08	BE	B0	0005F	MOVW	@LCLDSC+4, @KEYADR	..	0743	
		04	AE	9F	00064	PUSHAB	LCLDSC	..	0745	
		00000000	00	9F	00067	PUSHAB	P.AAY	..	0747	
		04	AC	DD	0006D	PUSHL	FILEID	..	0746	
			01	DD	00070	PUSHL	#1	..	0745	
00000000G	00		04	FB	00072	CALLS	#4, NML\$LOGRECORDOP			
	50		01	D0	00079	MOVL	#1, R0	..	0750	
			04	0007C	2\$: RET			..	0752	

; Routine Size: 125 bytes, Routine Base: \$CODE\$ + 02FC

```
761 0753 1 %SBTTL 'NMA$WRITEREC Write a Record to a File'
762 0754 1 GLOBAL ROUTINE NMA$WRITEREC (FILEID, KEYADR, BUFDSC) =
763 0755 1
764 0756 1 ++
765 0757 1 FUNCTIONAL DESCRIPTION:
766 0758 1
767 0759 1     This routine puts a record to the specified file. The key is
768 0760 1     specified by keyadr. The file was opened so that puts to existing
769 0761 1     records act as updates. The keyvalue is moved to the first two bytes
770 0762 1     of the record before the write.
771 0763 1
772 0764 1 FORMAL PARAMETERS:
773 0765 1
774 0766 1     FILEID      Value if the fileid
775 0767 1     KEYADR      Address of a word of keyvalue
776 0768 1     BUFDSC      Address of descriptor of data to write
777 0769 1
778 0770 1 IMPLICIT INPUTS:
779 0771 1
780 0772 1     NONE
781 0773 1
782 0774 1 IMPLICIT OUTPUTS:
783 0775 1
784 0776 1     NONE
785 0777 1
786 0778 1 ROUTINE VALUE:
787 0779 1 COMPLETION CODES:
788 0780 1
789 0781 1     RMS error code
790 0782 1
791 0783 1 SIDE EFFECTS:
792 0784 1
793 0785 1     NONE
794 0786 1
795 0787 1 --
796 0788 1
797 0789 2 BEGIN
798 0790 2
799 0791 2 MAP
800 0792 2     BUFDSC : REF VECTOR;           ! User supplied data
801 0793 2
802 0794 2 LOCAL
803 0795 2     RAB      : REF BLOCK [1, BYTE], ! Address of rab
804 0796 2     STATUS   : REF BLOCK [1, BYTE], ! Return status
805 0797 2     FILEDSC  : REF BLOCK [1, BYTE], ! File descriptor address
806 0798 2     FIELD (FDSCFLDS),
807 0799 2     LCLDSC   : VECTOR [2];
808 0800 2
809 0801 2     STATUS = NMA$SELECTFILE (.FILEID,
810 0802 2     FILEDSC); ! Obtain file descriptor
811 0803 2
812 0804 2 IF NOT .STATUS
813 0805 2 THEN
814 0806 2     RETURN .STATUS;           ! Return the status
815 0807 2
816 0808 2     RAB = .FILEDSC [FDSCRAB]; ! Obtain the rab address
817 0809 2     LCLDSC [0] = .BUFDSC [0] + NML$K_PERM_KEYS_LEN;
817 0809 2     LCLDSC [1] = .BUFDSC [1] - NML$K_PERM_KEYS_LEN;
```



```

818      RAB [RAB$W_RSZ] = .LCLDSC [0];      ! User buffer to write
819      RAB [RAB$R_RBF] = .LCLDSC [1];
820
821      NMA$W_KEYBUF = .KEYADR;      ! Key value from user
822      (.LCLDSC [1])<0,16,0> = .NMA$W_KEYBUF; ! Move key to buffer for write
823
824      STATUS = $PUT (RAB = .RAB);      ! Put or update the record
825
826      IF .STATUS
827      THEN
828          NML$LOGRECORDOP (DBG$C FILEID,
829                          FILEID,
830                          $ASCII ('record written'),
831                          LCLDSC);
832
833      RETURN .STATUS
834
835      END;
```

.PSECT \$PLITS,NOWRT,NOEXE,2

```

6E 65 74 74 69 72 77 20 64 72 6F 63 65 72 0013C P.ABB: .ASCII \record written\
                                0014A .BLKB 2
                                0000000E 0014C P.ABA: .LONG 14
                                00000000 00150 .ADDRESS P.ABB
```

.EXTRN SYSS\$PUT

.PSECT \$CODE\$,NOWRT,2

```

                                000C 00000
53 00000000' 00 9E 00002 .ENTRY NMA$WRITEREC, Save R2,R3 : 0754
5E          0C C2 00009 MOVAB NMA$W_KEYBUF, R3
          SE DD 0000C SUBL2 #12, SP
          04 AC DD 0000E PUSHL SP : 0801
          CF 02 FB 00011 CALLS #2, NMA$SELECTFILE
          52 50 D0 00016 MOVL R0, STATUS
          4C 52 E9 00019 BLBC STATUS, 1$ : 0803
          50 6E D0 0001C MOVL FILEDSC, R0 : 0807
          51 0C A0 D0 0001F MOVL 12(R0), RAB
          50 0C AC D0 00023 MOVL BUFDSC, R0 : 0808
          04 AE 60 02 C1 00027 ADDL3 #2, (R0), LCLDSC
          08 AE A0 02 C3 0002C SUBL3 #2, 4(R0), LCLDSC+4 : 0809
          22 A1 04 AE B0 00032 MOVL LCLDSC, 34(RAB) : 0810
          28 A1 08 AE D0 00037 MOVL LCLDSC+4, 40(RAB) : 0811
          63 08 BC B0 0003C MOVL @KEYADR, NMA$W_KEYBUF : 0813
          08 BE 63 B0 00040 MOVL NMA$W_KEYBUF, 2LCLDSC+4 : 0814
          00000000G 00 01 FB 00046 PUSHL RAB : 0816
          52 50 D0 0004D CALLS #1, SYSS$PUT
          15 52 E9 00050 MOVL R0, STATUS
          04 AE 9F 00053 BLBC STATUS, 1$ : 0818
          00000000' 00 9F 00056 PUSHAB LCLDSC : 0820
          04 AC DD 0005C PUSHAB P.ABA : 0822
          01 DD 0005F PUSHL FILEID : 0821
          01 DD 0005F PUSHL #1 : 0820
```

NMAFILES
V04-000

File Routines for Network Management
NMA\$WRITEREC Write a Record to a File

L I
16-Sep-1984 00:42:37
14-Sep-1984 12:50:02

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[NML.SRC]NMAFILES.B32;1 Page 26 (9)

00000000G 00
50

04 FB 00061
52 D0 00068 1\$:
04 0006B

CALLS #4, NML\$LOGRECORDOP
MOVL STATUS, R0
RET

: 0825
: 0827

; Routine Size: 108 bytes, Routine Base: \$CODE\$ + 0379

NML
V04

; R

```
037 0828 1 %SBTTL 'NMA$DELETEREC Delete a Record from the File'
038 0829 1 GLOBAL ROUTINE NMA$DELETEREC (FILEID, KEYADR) =
039 0830 1
040 0831 1 ++
041 0832 1 FUNCTIONAL DESCRIPTION:
042 0833 1
043 0834 1 This routine deletes a record from the file by specified key
044 0835 1 number.
045 0836 1
046 0837 1 FORMAL PARAMETERS:
047 0838 1
048 0839 1 FILEID Value if the fileid
049 0840 1 KEYADR Address of a word of keyvalue
050 0841 1
051 0842 1 IMPLICIT INPUTS:
052 0843 1
053 0844 1 NONE
054 0845 1
055 0846 1 IMPLICIT OUTPUTS:
056 0847 1
057 0848 1 NONE
058 0849 1
059 0850 1 ROUTINE VALUE:
060 0851 1 COMPLETION CODES:
061 0852 1
062 0853 1 RMS error code
063 0854 1
064 0855 1 SIDE EFFECTS:
065 0856 1
066 0857 1 NONE
067 0858 1
068 0859 1 --
069 0860 1
070 0861 1 BEGIN
071 0862 1
072 0863 1 LOCAL
073 0864 1 RAB : REF BLOCK [1, BYTE], ! Address of rab
074 0865 1 STATUS, ! Return status
075 0866 1 FILEDSC : REF BLOCK [1, BYTE] ! File descriptor address
076 0867 1 FIELD (FDSCFLDS);
077 0868 1
078 0869 1 STATUS = NMA$SELECTFILE (.FILEID,
079 0870 1 FILEDSC); ! Obtain file descriptor
080 0871 1
081 0872 1 IF .STATUS
082 0873 1 THEN
083 0874 1 BEGIN
084 0875 1
085 0876 1 RAB = .FILEDSC [FDSCRAB]; ! Obtain the rab address
086 0877 1
087 0878 1 NMA$W_KEYBUF = ..KEYADR; ! Key value from user
088 0879 1
089 0880 1 STATUS = $DELETE (RAB = .RAB); ! Delete the record
090 0881 1
091 0882 1 IF .STATUS
092 0883 1 THEN
093 0884 1 NML$LOGRECORDOP (DBG$C_FILEIO,
```



```
: 894      0885      2
: 895      0886      2
: 896      0887      2
: 897      0888      2
: 898      0889      2
: 899      0890      2
: 900      0891      2
: 901      0892      2
: 902      0893      1

      END;
      RETURN .STATUS
      END;
```

```
.FILEID,
$ASCII ('record deleted'),
UPLIT (2, NMA$W_KEYBUF));
```

.PSECT \$PLITS\$,NOWRT,NOEXE,2

```
64 65 74 65 6C 65 64 20 64 72 6F 63 65 72 00154 P.ABD: .ASCII \record deleted\
                                00162 .BLKB 2
                                0000000E 00164 P.ABC: .LONG 14
                                00000000' 00168 .ADDRESS P.ABD
                                00000002 0016C P.ABE: .LONG 2
                                00000000' 00170 .ADDRESS NMA$W_KEYBUF
```

.EXTRN SYSS\$DELETE

.PSECT \$CODE\$,NOWRT,2

```
                                0004 00000
                                04 C2 00002
                                5E DD 00005
                                04 AC DD 00007
                                FC8D CF 02 FB 0000A
                                52 50 D0 0000F
                                36 52 E9 00012
                                50 6E D0 00015
                                50 0C A0 D0 00018
                                00000000' 00 08 BC B0 0001C
                                00000000G 00 01 DD 00024
                                52 50 D0 00026
                                18 52 D0 0002D
                                00000000' 00 9F 00033
                                00000000' 00 9F 00039
                                04 AC DD 0003F
                                01 DD 00042
                                00000000G 00 04 FB 00044
                                50 52 D0 0004B 1$:
                                04 0004E

.ENTRY NMA$DELETEREC, Save R2
SUBL2 #4, SP
PUSHL SP
PUSHL FILEID
CALLS #2, NMA$SELECTFILE
MOVL R0, STATUS
BLBC STATUS, 1$
MOVL FILEDSC, R0
MOVL 12(R0), RAB
MOVW @KEYADR, NMA$W_KEYBUF
PUSHL RAB
CALLS #1, SYSS$DELETE
MOVL R0, STATUS
BLBC STATUS, 1$
PUSHAB P.ABE
PUSHAB P.ABC
PUSHL FILEID
PUSHL #1
CALLS #4, NML$LOGRECORDOP
MOVL STATUS, R0
RET
```

; Routine Size: 79 bytes, Routine Base: \$CODE\$ + 03E5

NMAFILES	File Routines for Network Management	B 2	
V04-000	NMA\$DELETEREC Delete a Record from the File	16-Sep-1984 00:42:37	VAX-11 Bliss-32 V4.0-742
		14-Sep-1984 12:50:02	DISK\$VMSMASTER:[NML.SRC]NMAFILES.B32;1 (11)
: 904	0894 1 END	! End of module	
: 905	0895 1		
: 906	0896 0 ELUDOM		

PSECT SUMMARY							
Name	Bytes	Attributes					
\$OWNS	1352	NOVEC,	WRT,	RD	NOEXE,NOSHR,	LCL, REL,	CON,NOPIC,ALIGN(2)
\$SPLITS	372	NOVEC,NOWRT,	RD	NOEXE,NOSHR,	LCL, REL,	CON,NOPIC,ALIGN(2)	
\$CODES	1076	NOVEC,NOWRT,	RD	EXE,NOSHR,	LCL, REL,	CON,NOPIC,ALIGN(2)	

Library Statistics					
File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
-\$255\$DUA28:[NML.OBJ]NMLLIB.L32;1	341	3	0	27	00:00.1
-\$255\$DUA28:[SHRLIB]NMALIBRY.L32;1	887	14	1	47	00:00.2
-\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	141	1	581	00:02.2

```

:
:
:      COMMAND QUALIFIERS
:
:      BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:NMAFILES/OBJ=OBJ$:NMAFILES MSRC$:NMAFILES/UPDATE=(ENH$:NMAFILES)
:
: Size:      1076 code + 1724 data bytes
: Run Time:   00:30.1
: Elapsed Time: 01:12.0
: Lines/CPU Min: 1784
: Lexemes/CPU-Min: 31149
: Memory Used: 196 pages
: Compilation Complete

```


0280

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

0281 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

